REMARKS

The claims are rejected under Section 112, first paragraph, for failing to comply with the enablement requirement. The office action indicates that a code address is only indicating a memory location, not the memory region or sub-region. But, even if this were so, a code address that has sub-regions makes perfect sense. In other words, it has a first sub-region and a second sub-region. This is shown in Figure 2, block 90. As explained in the specification, the core virtual machine may limit the virtual scope within a local memory sub-region of the code address 45. It is further explained at page 4, lines 18-22, that the core machine 25 may maintain a limited set of methods for which codes are allocated within the local memory sub-region for each smaller distributed version of the global memory lookup table. One such memory sub-region, namely, block 100, is shown in Figure 3. The global memory lookup table is divided into distributed lookup tables 110, shown in Figure 3.

In other words, there is no reason to read "receiving a code address including first and second local memory sub-regions" as saying that the address must have memory. It just means the code address must have sub-regions. Thus, it is respectfully submitted that the claim is being misread and reconsideration is requested.

It is believed that the Section 112 issue has adversely affected the consideration of the patentability of the claims. The claims call for a code address having two different sub-regions, each sub-region being associated with a first or second version of a smaller and distributed global method lookup table.

The office action, in paragraph 4, first points out that code address could be one memory address of the code. It is simply unclear what significance this statement has.

The Examiner then states it is unclear how the code address can include multiple memory sub-regions. This is explicitly explained in the specification. In other words, the address has two regions. The requirement that the sub-region has to contain the starting address and an ending address seems to be unsupported and it is in no way inconsistent with the claim language.

Previously, it was pointed out that the code address was broken up into local memory sub-regions, which is set forth in the claim and is not addressed in the office action. It was never argued that the code address must be broken up into memory portions.

There is no breaking of any code address into regions in the cited art, nor is there any breaking up of any code address into regions linked to first and second versions of a global method lookup table, respectively. This has never been pointed out and there is no support for the rejection.

For example, even if Matula did partition a large direct lookup table and a smaller direct lookup table, and those lookup tables are associated with memory regions, there is no suggestion, nor any position set forth in any office action to date, that those so-called memory sub-regions are actually sub-regions of the code address, not separate memory areas. Therefore, the office action fails to set out a *prima facie* rejection and reconsideration would be appropriate.

Respectfully submitted,

Date: January 30, 2009

[illegible]

Timothy N. Trop, Reg. No. 28,994
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
713/468-8880 [Phone]
713/468-8883 [Fax]

Attorneys for Intel Corporation